AMENDMENTS TO THE CLAIMS

Please amend claims 1, 11, and 21, as shown below. This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for allocating memory in a computer system, the method comprising:

outputting a request from an application to an operating system for allocation of a block of memory by the operating system to the application;

accessing the block of memory for the application;

dividing the block of memory into a plurality of frames, with each of the plurality of frames operable to store an indexing structure associated with an attribute of a data record;

dividing each of the frames into a plurality of instances, with each of the plurality of instances operable to store an index node of the indexing structure; and

associating ~~an application-defined instance type~~ the attribute with the plurality of instances for data storage using the plurality of instances.

2. (Original) The method of claim 1 wherein accessing the block of memory includes allocating a block of memory that begins on a page boundary.

3. (Original) The method of claim 1 wherein the size of the block of memory is determined by a coding parameter associated with the application.

4. (Original) The method of claim 1 wherein dividing the block of memory into frames includes identifying a first page boundary within the block of memory.

5. (Original) The method of claim 4 wherein dividing the block of memory into frames further includes designating a portion of the block of memory before the first page boundary as unused.

6. (Original) The method of claim 1 wherein a size of each frame is determined by a coding parameter.

7. (Original) The method of claim 1 wherein a size of each frame is determined by a page size used by the operating system.

8. (Original) The method of claim 1 wherein dividing a block of memory into frames includes:

determining a last page boundary within the block of memory; and

designating a portion of the block of memory after the last page boundary as unused.

9. (Original) The method of claim 1 wherein a single type of data is stored in the block of memory.

10. (Original) The method of claim 1 wherein data from a fast query system is stored in the instances.

11. (Currently Amended) A software application tangibly embodied on a computer-readable medium using application-level memory management, the software application comprising:

an application-level memory manager operable to allocate a block of memory to store data elements, ~~divide~~ to divide the block of memory into a plurality of frames, with each of the plurality of frames operable to store an indexing structure associated with an attribute of a data record, and further to divide each frame into a plurality of instances, with each of the plurality of instances operable to store an index node of the indexing structure; and

application code operable to ~~define data elements as having an instance type, and to~~ associate the ~~instance type~~ attribute with the plurality of instances for storage of the data elements in the plurality of instances.

12. (Original) The software application of claim 11 wherein the block of memory begins on a page boundary.

13. (Original) The software application of claim 11 wherein the size of the block of memory is determined by a coding parameter.

14. (Original) The software application of claim 11 wherein the block of memory includes a first page boundary.

15. (Original) The software application of claim 14 wherein a portion of the block of memory before the first page boundary is designated as unused.

16. (Original) The software application of claim 11 wherein a size of each frame is determined by a coding parameter.

17. (Original) The software application of claim 11 wherein a size of each frame is determined by the page size used by the operating system.

18. (Original) The software application of claim 11 wherein the block of memory includes a last page boundary and a portion of the block of memory after the last page boundary is designated as unused.

19. (Original) The software application of claim 11 wherein a single type of data is stored in the block of memory.

20. (Original) The software application of claim 11 wherein the application code implements a fast query system.

21. (Currently Amended) A method comprising:

associating data elements used by an application with an application-defined instance type;

associating the application-determined instance type with an application-determined one of a plurality of blocks of memory allocated by an operating system, wherein the application-determined memory block is divided into a plurality of frames, with each of the plurality of frames operable to store an indexing structure associated with an attribute of a data record, the plurality of frames being ~~that are~~ further divided into a plurality of instances, with each of the plurality of instances operable to store an index node of the indexing structure; and

populating the plurality of instances with the data elements.

22. (Original) The method of claim 21 wherein associating the application-determined instance type with the application-determined block of memory comprises associating a single application-determined instance type with the application-determined block of memory.

23. (Original) The method of claim 21 further comprising:

removing the data elements; and

returning the block of memory to the operating system.

24. (Original) The method of claim 23 wherein returning the block of memory to the operating system comprises:

returning the block of memory to a buffer; and

determining after a pre-determined period of time that the block of memory is no longer required by the application.